Fonctions logiques



Des « 1 » et des « 0 »

- Systèmes électroniques : communication par chaîne de BITs (Binary digiT)
- L'information
  - Sa représentation : 0, 1
  - Son traitement : manipulation des 0 et des 1
- Réalisation physique de « 0 » et de « 1 »
  - électronique (tension,courant), optique (puissance optique, polarisation)
  - mécanique (levier, poulies), méca. flu. (fluides, vanne…)
  - ADN, physique quantique (spin)
  - objets courants : fiches perforées, pièces de monnaie (pile/face)



Des zéros, des uns et des portes



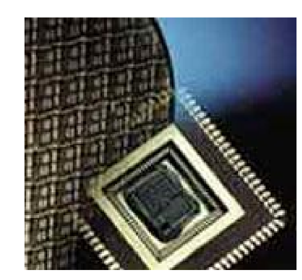
- Signaux numériques : 0 1
  - Nécessité d'une algèbre
  - Chaque système a sa propre algèbre
  - Monde numérique : 2 valeurs → algèbre booléenne (commutation)
    - Boole : 1850
    - Shannon : 1938
  - Variable/fonction logique
- Une porte : brique élémentaire de tout système numérique
  - réalise des fonctions utiles
  - possibilité de combinaisons pour des fonctions **encore plus utiles**
  - Quelques outils mathématiques pour nous aider

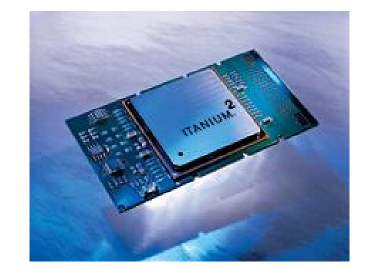


Portes logiques



- Elle nous sert à manipuler les 0 et les 1
- Petit composant électronique permettant de réaliser des fonctions simples
  - Portes de base : NOT, AND, OR, NAND et NOR
- Porte : constituées de plusieurs transistors
- Combinaison de portes : fonctions plus complexes
  - multiplieur, additionneur, soustracteur, ...







La table de vérité

Une ***table de vérité*** nous fait connaître la réaction d'un circuit logique (sa valeur de sortie) aux diverses combinaisons de niveaux logiques appliqués aux entrées (2^n).

Table 4 -4
General truth table structure for a 3-variable logic function, F(X, Y, Z).

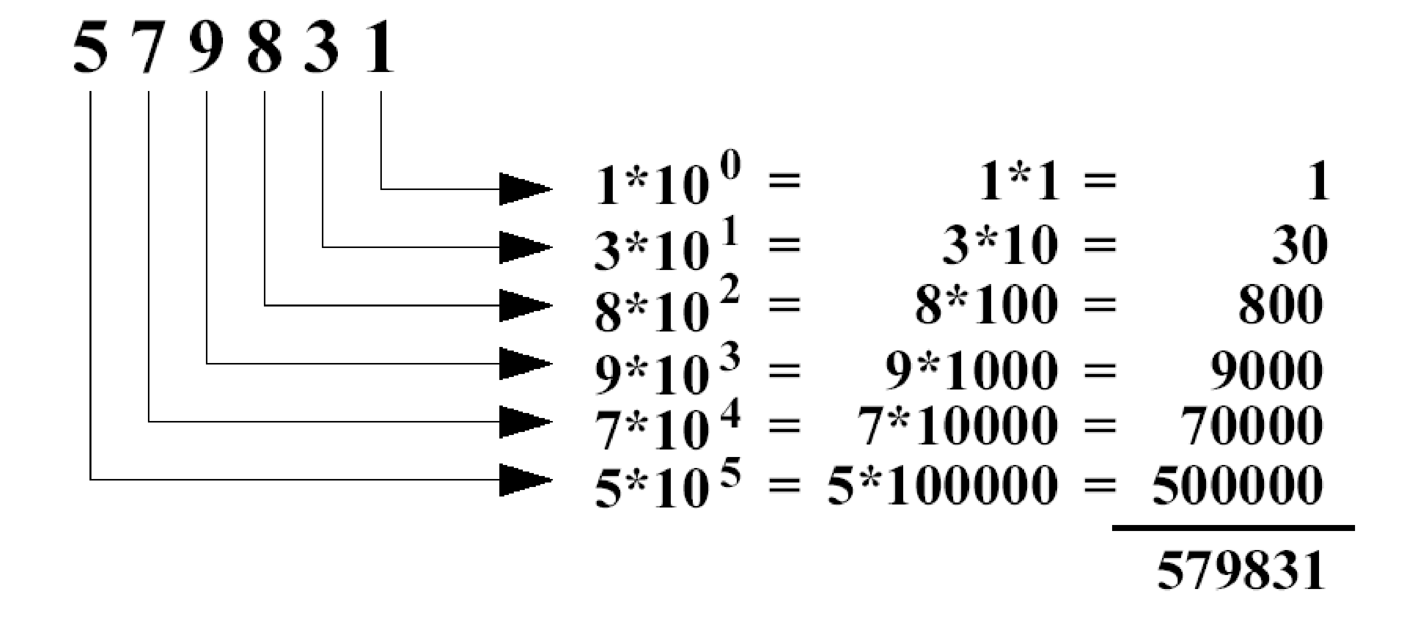
Row	X	Y	Z	F
0	0	0	0	F(0,0,0)
1	0	0	1	F(0,0,1)
2	0	1	0	F(0,1,0)
3	0	1	1	F(0,1,1)
4	1	0	0	F(1,0,0)
5	1	0	1	F(1,0,1)
6	1	1	0	F(1,1,0)
7	1	1	1	F(1,1,1)

Table 4 -5
Truth table for a particular 3-variable logic function, F(X, Y, Z).

Row	X	Y	Z	F
0	0	0	0	1
1	0	0	1	0
2	0	1	0	0
3	0	1	1	1
4	1	0	0	1
5	1	0	1	0
6	1	1	0	1
7	1	1	1	1

Représentation des nombre (1/3)

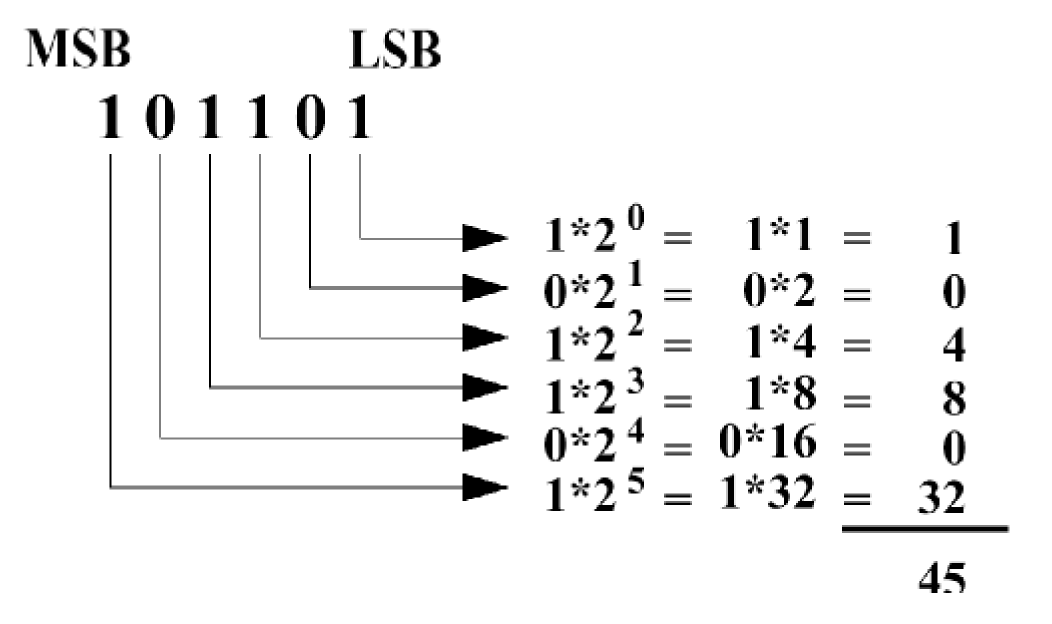
- En base de 10 : utilisée tous les jours





Représentation des nombre (2/3)

- En base de 2 : utilisée aussi tous les jours!



- Un truc pour coder en binaire ?



L'octet

- L'octet : 8 bits = 1 byte
- Le mot (word) : 16 bits
- Le double mot (double word ou dword) : 32 bits
- Des kilos, des méga
  - 1ko = 1024 octets
  - 1998 : standardisation par l 'IEC
    - 1 ko = 1000 octets
    - des kibi (Kio), mebi (Mio), gibi (Gio)
    - IEC : http://physics.nist.gov./cuu/Units/binary.html



Représentation des nombre (3/3)

- Autres codes : BCD (Binary Coded Decimal), code Gray
- Avec ou sans signe??
  - Valeur absolue/signée
  - Complément à 1 ou restreint
  - Complément à 2 ou complément vrai
  - Complément à excédent 2^m
- Avec ou sans virgule??
  - Nombre à virgule fixe
  - Nombre à virgule flottante



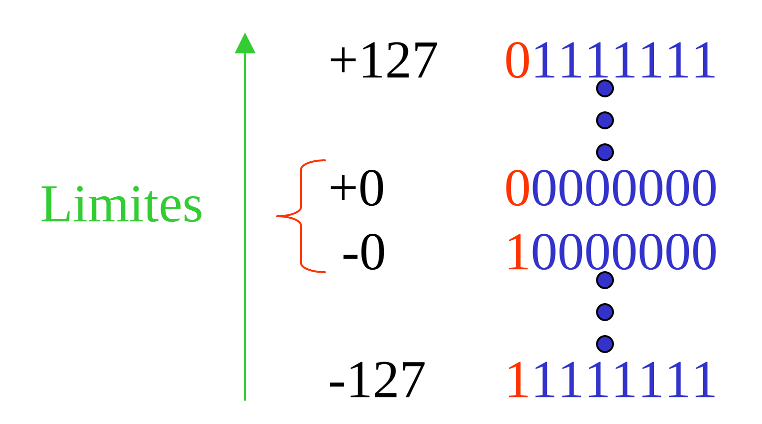
Le signe (1/2)

Valeur absolue et signée

Principe +11 00001101

-11 10001101

Limites

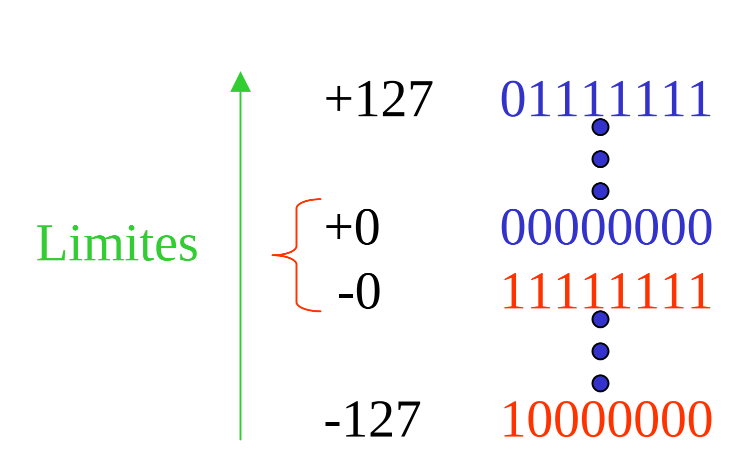


Complément à 1

Principe +11 00001101

-11 11110010

Limites



Le signe (2/2)

Complément à 2

Principe

+11		00001101
		11110010
	+	00000001
-11		11110011

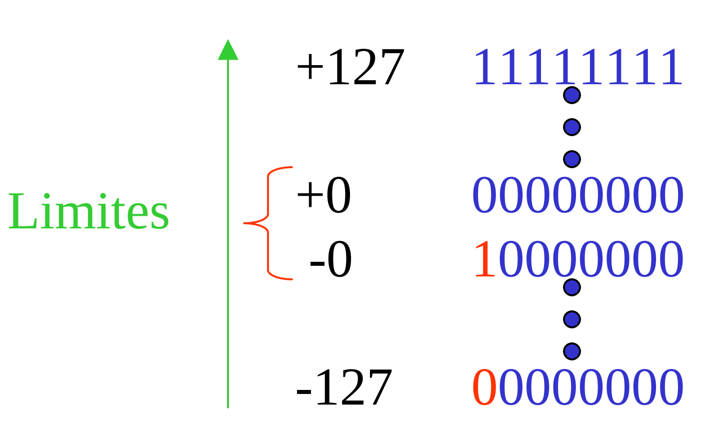
Limites

+127	01111111
+0 / -0	00000000
-128	10000000

Complément à 2^m (m=7)

Principe

+11		00001101
		11110011
2^7	+	10000000
-11		01110011

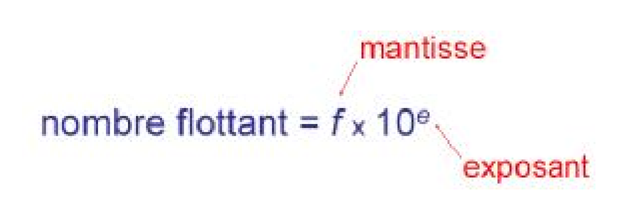


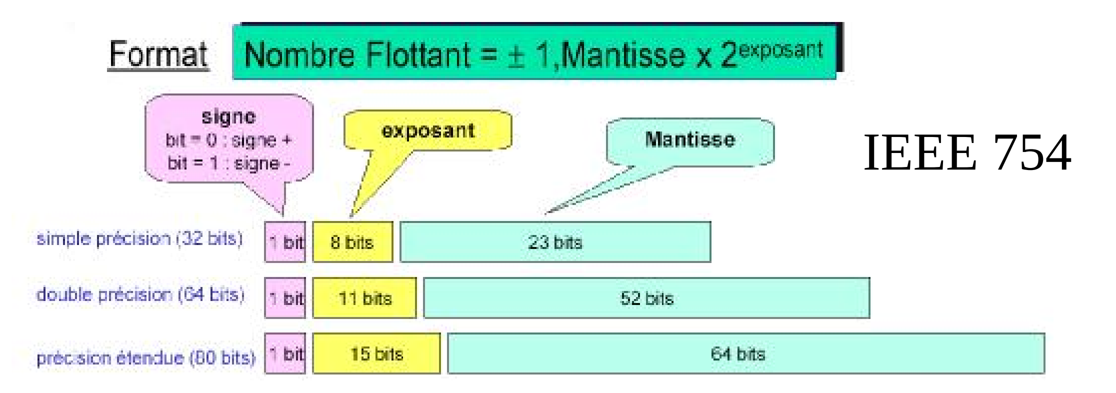
Des virgules

- La virgule fixe



- La virgule flottante





IEEE 754



Les opérations

- L’addition : sans problème OU/OU exclusif
- La multiplication : sans problème
  - ET
  - décalage
  - addition
- La soustraction
  - logique signée
  - Complément à 2
  - addition
- La division

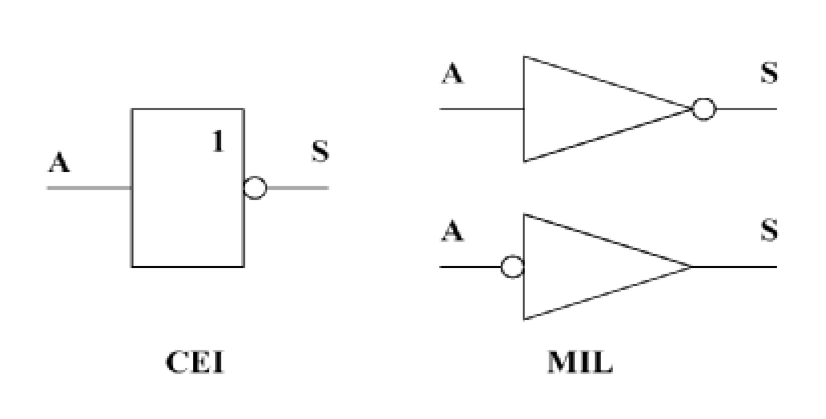


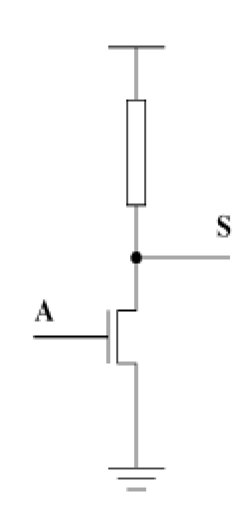
Les portes

- Porte inverseuse ou NON (NOT)
- Porte NON-ET (NAND)
- Porte NON-OU (NOR)
- Porte ET (AND)
- Porte OU (OR)
- Symboles, table de vérité et implémentation physique
- Porte de transmission et buffer



La porte NON/INVERSEUSE





- CEI : Commission Électrotechnique Internationale
- MIL : MILitary standards (US department of Defense)
- Table de vérité

A	S
0	1
1	0

$S = \bar{A}$



Porte NON & technos

MOSFET

Bipolaire

RTL

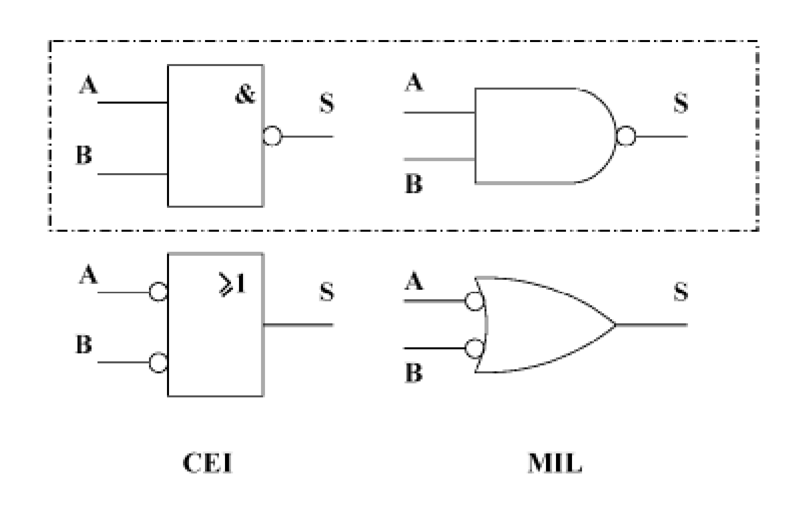
TTL

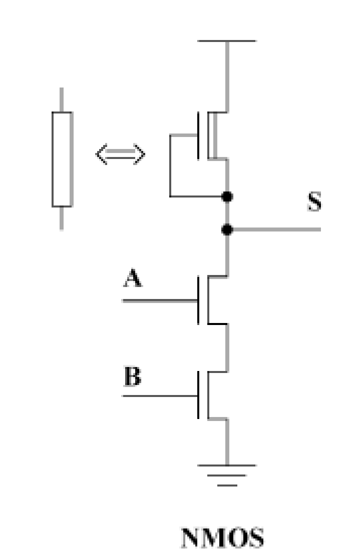
ECL

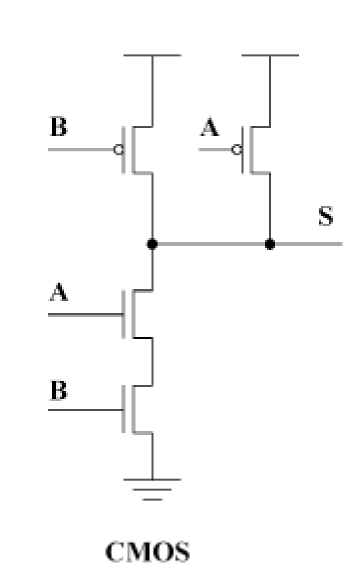
- MOS : NMOS (voire NMOS déplétée) ou CMOS
- Bipolaire
  - RTL : Resistor Transistor Logic
  - TTL : Transistor Transistor Logic
  - ECL : Emitter Coupled/Collector Logic



Porte NON-ET (NAND)







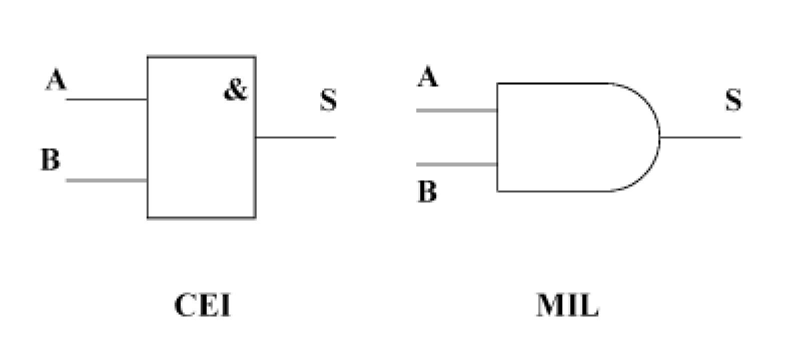
- Table de vérité :

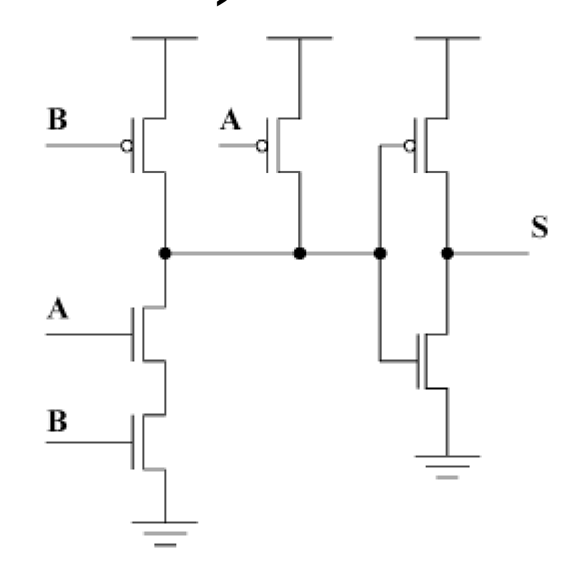
A	B	S
0	0	1
0	1	1
1	0	1
1	1	0

$$S = \overline{A.B} = \bar{A} + \bar{B}$$



Porte ET (AND)





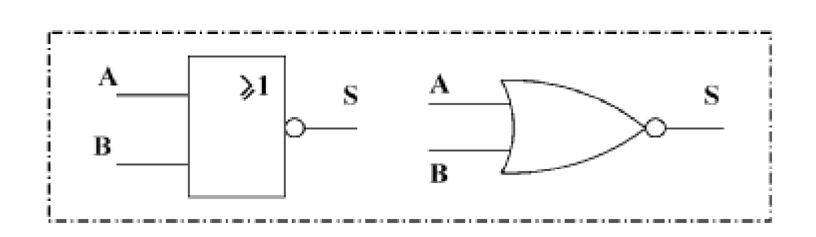
- Table de vérité :

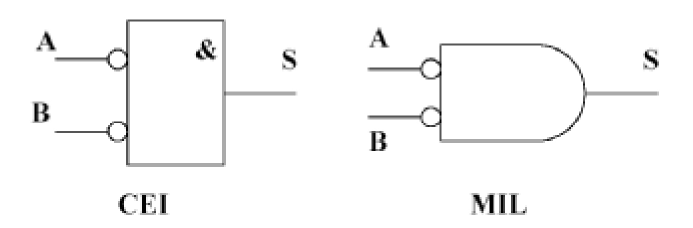
A	B	S
0	0	0
0	1	0
1	0	0
1	1	1

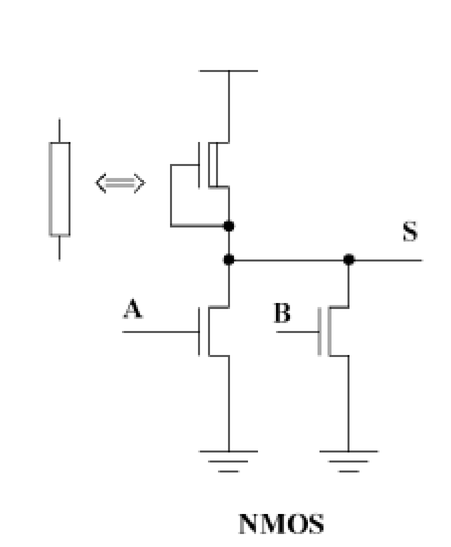
$S = A.B$

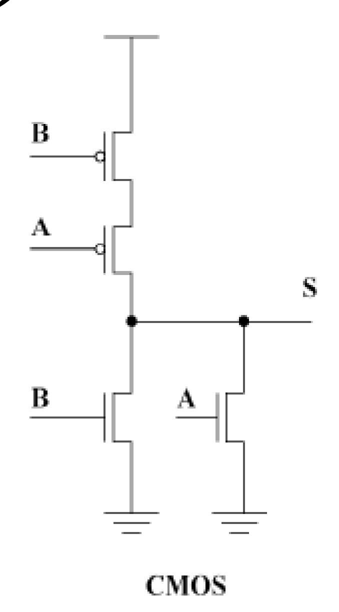


Porte NON-OU (NOR)









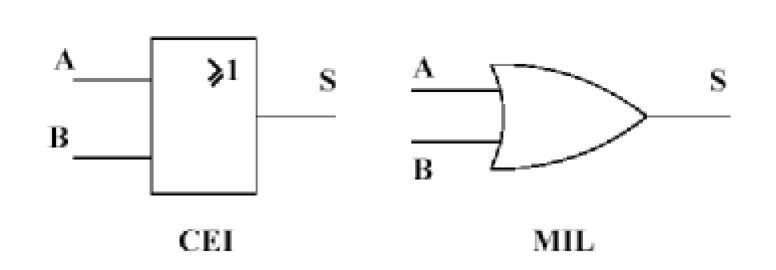
- Table de vérité :

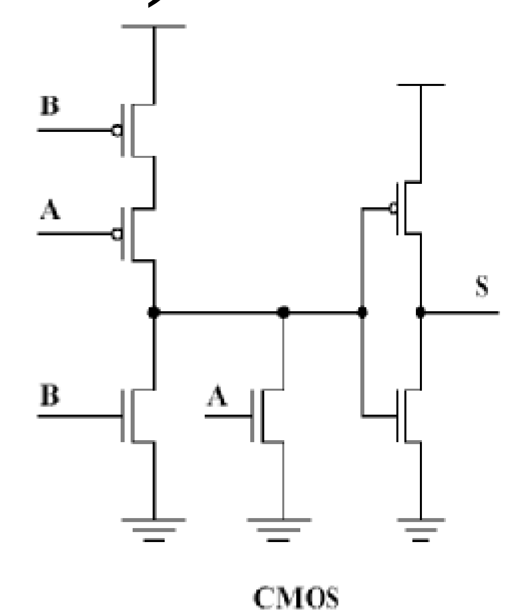
A	B	S
0	0	1
0	1	0
1	0	0
1	1	0

$$S = \overline{A + B} = \bar{A}.\bar{B}$$



Porte OU (OR)





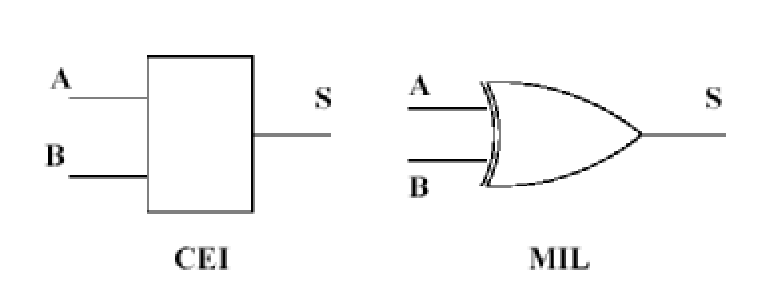
- Table de vérité :

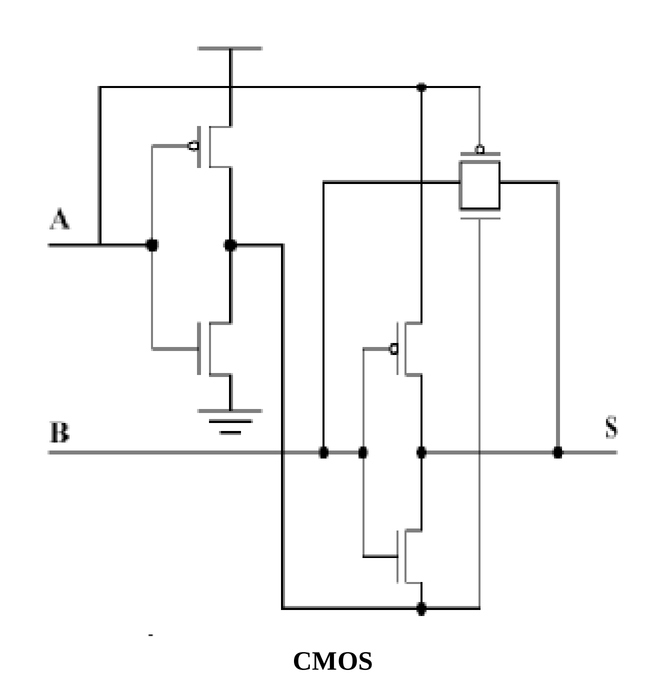
A	B	S
0	0	0
0	1	1
1	0	1
1	1	1

$$S = A + B$$



Porte OU-EXCLUSIF (XOR)





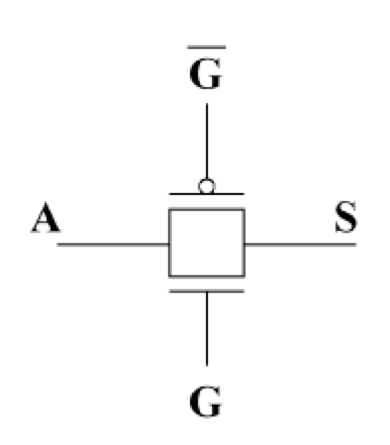
- Table de vérité :

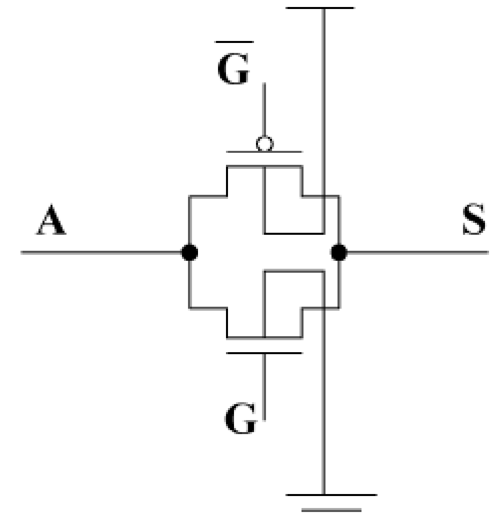
A	B	S
0	0	0
0	1	1
1	0	1
1	1	0

$$S = A \oplus B = \bar{A}.B + A.\bar{B}$$



Porte de transmission





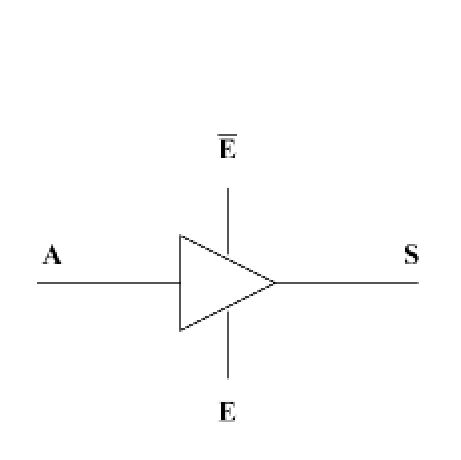
CMOS

- Table de vérité :

G	A	S
0	0	Z
0	1	Z
1	0	0
1	1	1



Circuit tampon (buffer)



CMOS

- Table de vérité :

E	A	S
0	0	Z
0	1	Z
1	0	0
1	1	1

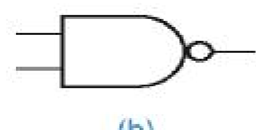
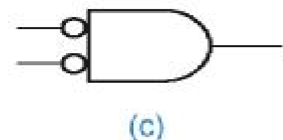
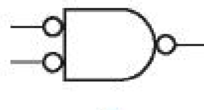


Équivalence de portes (1/2)

Copyright © 2000 by Prentice Hall, Inc.
Digital Design Principles and Practices, 3/e

Figure 5-6 (Fonction ET)

a) 7408 porte AND
b) 7400 porte NAND
c) 7402 porte NOR
d) 7432 porte OR

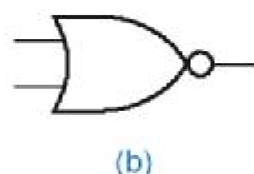
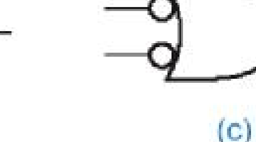
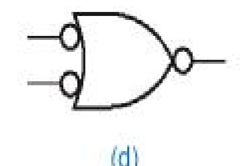




(a) (b) (c) (d)

Figure 5-7 (Fonction OU)

Copyright © 2000 by Prentice Hall, Inc.
Digital Design Principles and Practices, 3/e

a) 7432 porte OR
b) 7402 porte NOR
c) 7400 porte NAND
d) 7408 porte AND

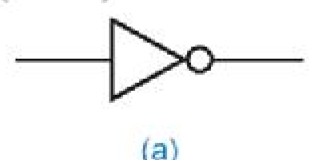
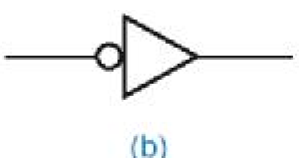
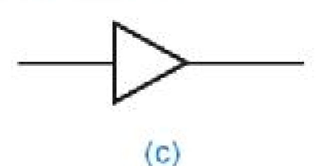
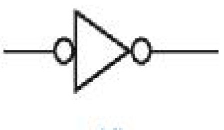
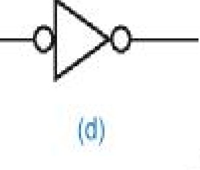




(a) (b) (c) (d)

Figure 5-7 (Fonction INV)

a) b) Inverseur
c) d) Suiveur (buffer)

Copyright © 2000 by Prentice Hall, Inc.
Digital Design Principles and Practices, 3/e

(a) (b) (c) (d)

Équivalence de portes (2/2)

(a)

≡ ≡ ≡

(b)

≡ ≡ ≡

Copyright © 2000 by Prentice Hall, Inc.
Digital Design Principles and Practices, 3/e

Figure 5-72a (Fonction XOR)

74x86 porte XOR

Figure 5-71b (Fonction XNOR)

74x266 porte XOR

Porte complète

- On appelle porte complète toute porte capable de réaliser toutes les opérations logiques
  - Fonction OU (Addition)
  - Fonction ET (Multiplication)
  - Fonction NON (Inversion)
- Les gagnants sont
  - NAND

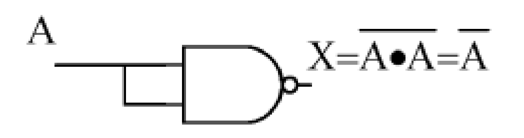
$$\bar{A} = A \downarrow A,\ A.B = \overline{A \downarrow B} \text{ et } A + B = \bar{A} \downarrow \bar{B}$$

  - NOR

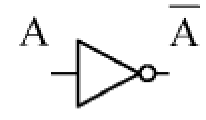
$$\bar{A} = A \uparrow A,\ A.B = \bar{A} \uparrow \bar{B} \text{ et } A + B = \overline{A \uparrow B}$$



Un exemple concret d'universalité



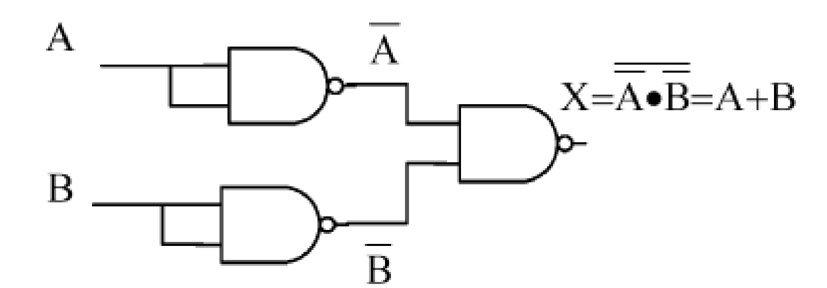
ce qui fonctionnellement équivaut à



A B $\overline{A \bullet B}$ X=A•B

ce qui fonctionnellement équivaut à

A B X=A•B

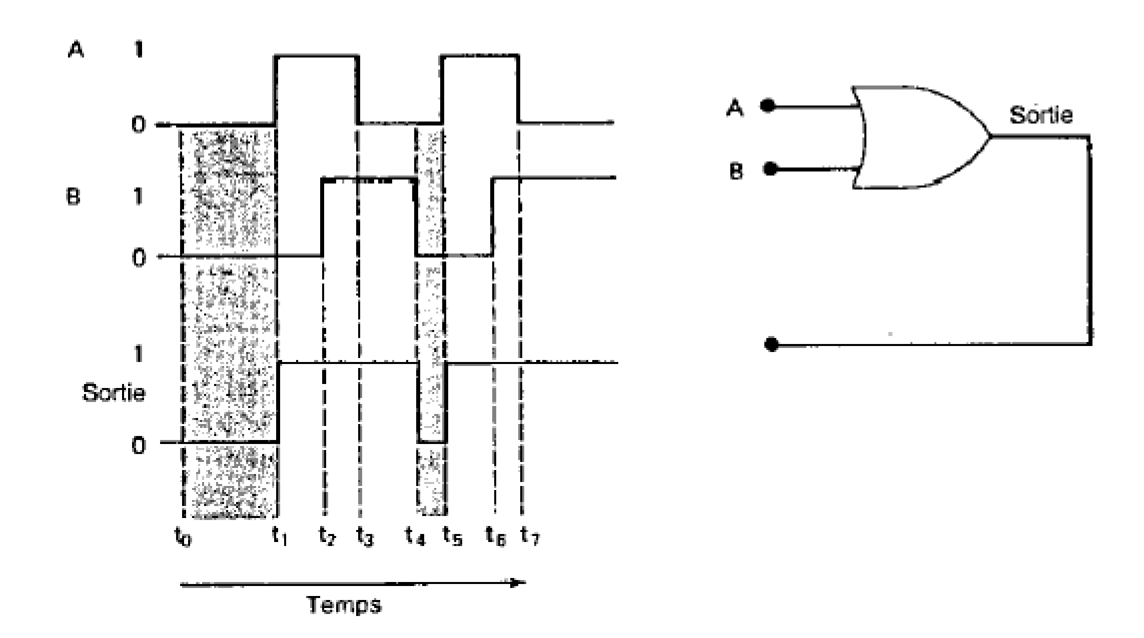


ce qui fonctionnellement équivaut à

A B X=A+B

Pour plus tard….

Ne pas oublier le chronogramme (diagramme temporel)



Pour cet exemple, le temps de propagation de la porte (t_p) = 0.



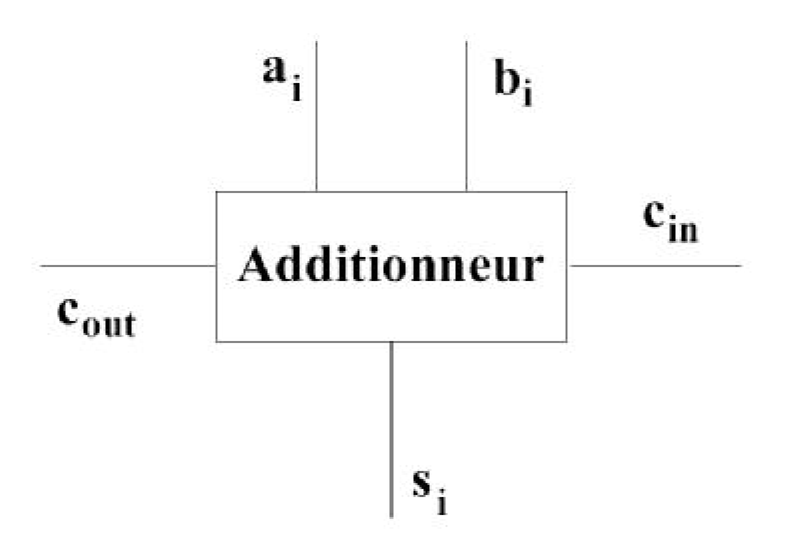
Modules arithmétiques

- La base de nombreuses applications : UAL (processeur!!)
- Nécessite l'implémentation de :
  - addition
  - multiplication
  - soustraction
  - division
- Brique de base : additionneur 2 bits



Additionneur 2 bits

- Symbole



- Table de vérité :

a	b	c_{in}	s	c_{out}
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Synthèse de fonctions combinatoires

- Consiste à spécifier les opérateurs matériels permettant l 'implémentation physique d 'une table de vérité ou d 'une expression booléenne
- 3 grandes méthodes selon la complexité des portes utilisées
  - logique anarchique : minimisation du nbre de portes à implémenter
  - logique structurée : accent sur le nbre E/S (ROM, RAM, …)
  - logique en tranche : réalisation d'opérateurs **n** bits à partir d'opérateur 1 bit (additionneur **n** bits, ...)
- Pour ce faire : une algèbre, ses règles et quelques outils



Quelques théorèmes

Table 4-1 Switching-algebra theorems with one variable.

(T1)	$X + 0 = X$	(T1′)	$X \cdot 1 = X$	(Identities)
(T2)	$X + 1 = 1$	(T2′)	$X \cdot 0 = 0$	(Null elements)
(T3)	$X + X = X$	(T3′)	$X \cdot X = X$	(Idempotency)
(T4)	$(X')' = X$			(Involution)
(T5)	$X + X' = 1$	(T5′)	$X \cdot X' = 0$	(Complements)

	$X + X' \cdot Y = X + Y$	
Ou exclusif	$= X \cdot Y' + X' \cdot Y$	
Non-Ou ex	$= X \cdot Y + X' \cdot Y'$	

Table 4-2 Switching-algebra theorems with two or three variables.

(T6)	$X + Y = Y + X$	(T6′)	$X \cdot Y = Y \cdot X$	(Commutativity)
(T7)	$(X + Y) + Z = X + (Y + Z)$	(T7′)	$(X \cdot Y) \cdot Z = X \cdot (Y \cdot Z)$	(Associativity)
(T8)	$X \cdot Y + X \cdot Z = X \cdot (Y + Z)$	(T8′)	$(X + Y) \cdot (X + Z) = X + Y \cdot Z$	(Distributivity)
(T9)	$X + X \cdot Y = X$	(T9′)	$X \cdot (X + Y) = X$	(Covering)
(T10)	$X \cdot Y + X \cdot Y' = X$	(T10′)	$(X + Y) \cdot (X + Y') = X$	(Combining)
(T11)	$X \cdot Y + X' \cdot Z + Y \cdot Z = X \cdot Y + X' \cdot Z$			(Consensus)
(T11′)	$(X + Y) \cdot (X' + Z) \cdot (Y + Z) = (X + Y) \cdot (X' + Z)$			

Table 4-3 Switching-algebra theorems with n variables.

(T12)	$X + X + \cdots + X = X$	(Generalized idempotency)
(T12′)	$X \cdot X \cdot \cdots \cdot X = X$	
(T13)	$(X_1 \cdot X_2 \cdot \cdots \cdot X_n)' = X_1' + X_2' + \cdots + X_n'$	(DeMorgan's theorems)
(T13′)	$(X_1 + X_2 + \cdots + X_n)' = X_1' \cdot X_2' \cdot \cdots \cdot X_n'$	
(T14)	$[F(X_1, X_2, \ldots, X_n, +, \cdot)]' = F(X_1', X_2', \ldots, X_n', \cdot, +)$	(Generalized DeMorgan's theorem)
(T15)	$F(X_1, X_2, \ldots, X_n) = X_1 \cdot F(1, X_2, \ldots, X_n) + X_1' \cdot F(0, X_2, \ldots, X_n)$	(Shannon's expansion theorems)
(T15′)	$F(X_1, X_2, \ldots, X_n) = [X_1 + F(0, X_2, \ldots, X_n)] \cdot [X_1' + F(1, X_2, \ldots, X_n)]$	

20



Tableau de Karnaugh

- Maurice Karnaugh : 1950 Bell Lab's

Maurice Karnaugh: "The Map Method for Synthesis of Combinational Logic Circuits', Transactions of the AIEE, Vol. 72, No. 9 (1953), 593-599

- Tous les termes pour lesquels la fonction est à 1 (0) doivent être pris au moins une fois dans un regroupement ou seuls si aucun regroupement n 'est possible
- Faire les regroupements de taille maximale, de manière à éliminer le plus grand nombre possible de variables dans les termes de l 'expression
- Ne prendre que les regroupements ou termes produits nécessaires pour prendre en compte au moins une fois chaque 1 (0) afin d 'éviter les redondances
- PS : binaire réfléchi….



Application : additionneur 2 bits

- La somme

c_{in} \ ab	00	01	11	10
0	0	1	0	1
1	1	0	1	0

- Raisonnons sur s_i : $s = 1$ quand (posons $c = c_{in}$).....
  - Pas de groupement possible
  - $s = \bar{a}.\bar{b}.c + \bar{a}.b.\bar{c} + a.b.c + a.\bar{b}.\bar{c}$
  - Des OU EXCLUSIFS !!
    - $s = (\bar{a}.\bar{b} + a.b).c + (\bar{a}.b. + a.\bar{b}).\bar{c}$
    - $s = (a \oplus b) \oplus c$



Application : additionneur 2 bits

- La retenue

c_{in} \ ab	00	01	11	10
0	0	0	1 (α)	0
1	0	1 (β)	1 (α, β, σ)	1 (σ)

- Raisonnons sur c_{out} : $c_{out} = 1$ quand.....
  - $c_{out} = a.b.\bar{c} + \bar{a}.b.c + a.b.c + a.\bar{b}.c$
  - En simplifiant ($a = a + a$ et $b + \bar{b} = 1$)
    - $c_{out} = a.b.\bar{c} + \bar{a}.b.c + a.b.c + a.\bar{b}.c$
    - $c_{out} = a.b.(\bar{c} + c) + \bar{a}.b.c + a.\bar{b}.c$
    - $c_{out} = a.b + \bar{a}.b.c + a.\bar{b}.c, \ldots$ il y a mieux
  - 3 groupements possibles : α, β, σ
    - $c_{out} = a.b + b.c + a.c$

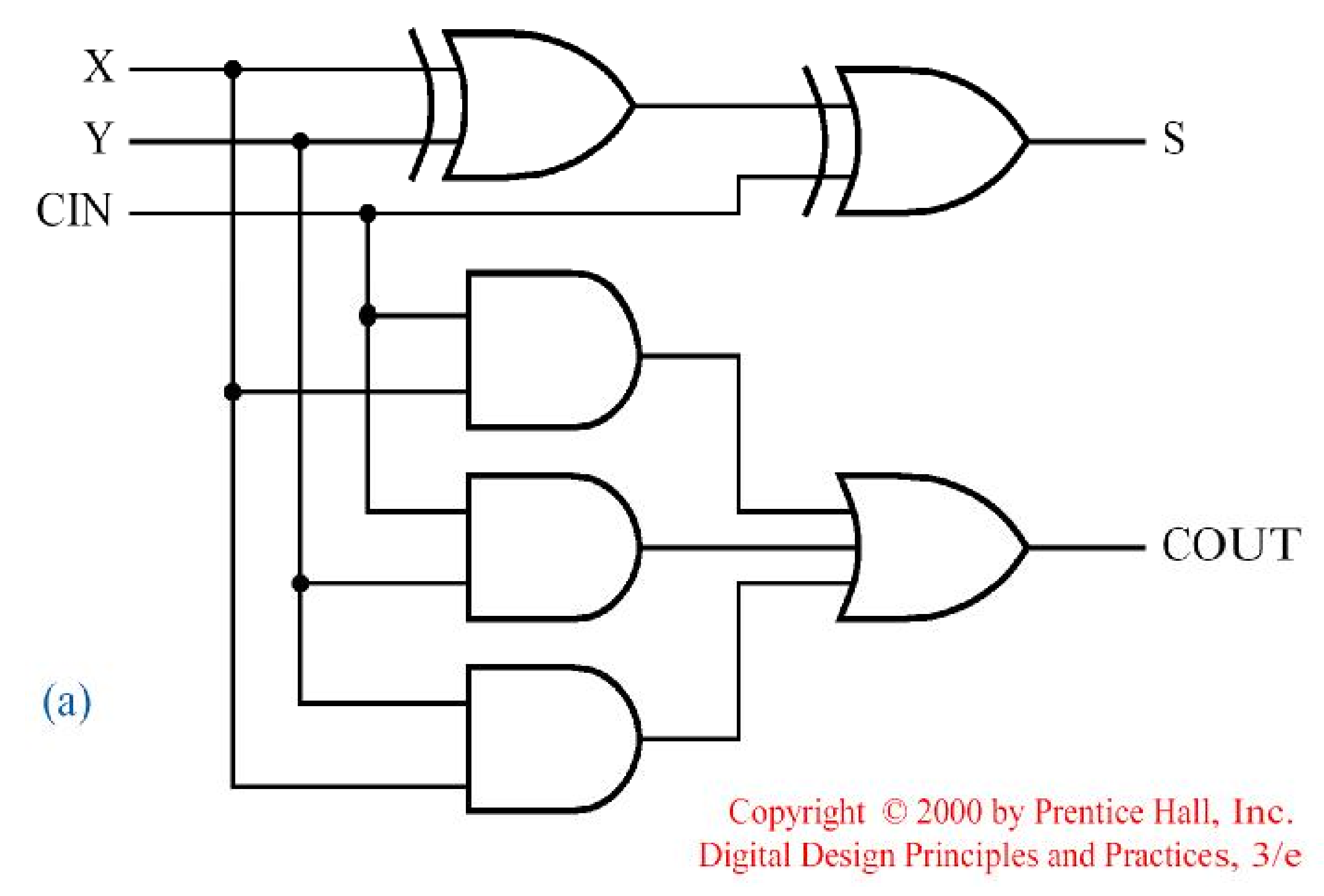


Additionneur 2 bits : synthèse

- Synthèse de la somme
  - Synthèse d'une porte OU-EXCLUSIF
  - $s = a \oplus b = \bar{a}.b + a.\bar{b} = \overline{\overline{\bar{a}.b + a.\bar{b}}}$
  - $s = \overline{\overline{\bar{a}.b}.\overline{a.\bar{b}}} = \overline{\bar{a}.b} \downarrow \overline{a.\bar{b}} = (\bar{a} \downarrow b) \downarrow (a \downarrow \bar{b})$
- Synthèse de la retenue
  - $c_{out} = a.b + b.c + a.c = \overline{\overline{a.b + b.c + a.c}} = \overline{\overline{a.b}.\overline{a.c}.\overline{b.c}}$
  - $c_{out} = \overline{(a \downarrow b).(a \downarrow c).(b \downarrow c)} = (a \downarrow b) \downarrow (a \downarrow c) + \overline{b \downarrow c}$
  - or $A + \bar{B} = \overline{\bar{A}.B} = \bar{A} \downarrow B$
  - d'où $c_{out} = \underbrace{\overline{(a \downarrow b) \downarrow (a \downarrow c)}}_{=A} \downarrow \underbrace{(b \downarrow c)}_{=B}$



Implémentation physique



En portes NAND

r_{i-1} r_{i-1}

a_i $\overline{a_i}$

b_i $\overline{b_i}$

s_i r_i

Améliorations possibles

- Anticipation de la retenue
  - Le résultat est délivré lorsque la dernière retenue est propagée
  - Perte de temps
  - Une étude de la table de vérité fait apparaître la possibilité de déclencher la retenue en avance
- Cellule de Brent & Kung
- Additionneur de Sklanski
- Additionneurs de Kogge et Stone
- Additionneurs de Ling



Les « Don't care »

- Considérons un détecteur de nombre premier (BCD)
- Table de vérité : S vaut 1 pour 1,2,3,5 et 7 (au-delà pas d'importance $\rightarrow$ BCD)
- Tableau de Karnaugh

ab \ cd	00	01	11	10
00	0	1	1	1
01	0	1	1	0
11	X	X	X	X
10	0	0	X	X

- Groupement α, β ou mieux α, σ
- $s = \bar{a}.\bar{b}.d + \bar{b}.c$ ou $s = \bar{a}.d + \bar{b}.c$

En résumé

- Porte de base : NOT, AND, NAND, OR, NOR, XOR + transmission et buffer
- Portes complètes : NAND et NOR
- Algèbre de Boole
- Tableaux de Karnaugh
- Synthèse de fonctions combinatoire



Le multiplieur

Délai : 4, activité : 5

01011000
01011000
00000000
01011000
00000000
00000000
01011000
00000000

A 88 X 75 P 6600 Nb. bits 8 VHDL

Bien évidemment on peut compliquer pour simplifier ….



L’anticipation de retenue

I)

$$r_i = a_i \cdot b_i + r_{i-1}(a_i + b_i) \qquad (1)$$

$$r_i = a_i \cdot b_i + r_{i-1}(a_i \oplus b_i)$$

En posant :

$$G_i = a_i \cdot b_i$$

et $P_i = a_i + b_i$ ou $P_i = a_i \oplus b_i$

G_i et P_i sont respectivement les **fonctions génération et propagation de retenue.** La retenue de sortie peut être réécrite sous la forme :

$$r_i = G_i + r_{i-1} \cdot P_i$$

$$S_i = a_i \oplus b_i \oplus r_{i-1} = G_i \oplus P_i \oplus r_{i-1} = \overline{G_i} \cdot P_i \oplus r_{i-1}$$

a_i	b_i	G_i	P_i	$a_i \oplus b_i$	$G_i \oplus P_i$
0	0	0	0	0	0
0	1	0	1	1	1
1	0	0	1	1	1
1	1	1	1	0	0

II)

$$G_{i+3,i} = G_{i+3} + P_{i+3}\,G_{i+2} + P_{i+3}\,P_{i+2}\,G_{i+1} + P_{i+3}\,P_{i+2}\,P_{i+1}\,G_i \qquad (3)$$

$$P_{i+3,i} = P_{i+3}\,P_{i+2}\,P_{i+1}\,P_i \qquad (4)$$

$$r_{i+3} = G_{i+3} + P_{i+3}\,G_{i+2} + P_{i+3}\,P_{i+2}\,G_{i+1} + P_{i+3}\,P_{i+2}\,P_{i+1}\,G_i + P_{i+3}\,P_{i+2}\,P_{i+1}\,P_i\,r_{i-1} \qquad (2)$$

$$r_{i+3} = G_{i+3,i} + P_{i+3,i}\,r_{i-1}$$

